International Journal of Power Electronics and Drive System (IJPEDS)
Vol. 8, No. 1, March 2017, pp. 167~175
ISSN: 2088-8694, DOI: 10.11591/ijpeds.v8i1.pp167-175

Design and Implementation of Pulse Width Modulation Using Hardware/Software MicroBlaze Soft-Core

A. Zemmouri[1], R. Elgouri[2], M. Alareqi[3], M. Benbrahim[4], L. Hlou[5]
[1-5] Laboratory of Electrical Engineering and Energy System. Faculty of Sciences, University Ibn Tofail Kenitra, Morocco
[2,4] Laboratory of Electrical Engineering and Telecommunications Systems, National Schools of Applied Sciences, University Ibn Tofaïl Kenitra, Morocco
[3] Community College, Sana'a, Yemen

Article Info

Article history:

Received Oct 04, 2016
Revised Dec 10, 2016
Accepted Dec 20, 2016

Keyword:

EDK/SDK
FPGA Vertix5
Micoblaze
PWM

ABSTRACT

This paper presents an embedded control application of clock frequency to control the pulse width of the output signals, implemented on field programmable get array. This control allows the creation of lines of Pulse-width modulation depending on the numbers of card outputs, without using the specific "Timers /Counters" blocks; this method is effective to adjust the amount of power supplied to an electrical charge. The purpose of this work is to achieve a real time hardware implementation with higher performance in both size and speed. Performance of these designimplemented in field programmable get array virtex 5 card, and signals displayed on an asciloscope.

Copyright © 2017 Institute of Advanced Engineering and Science.
All rights reserved.

Corresponding Author:

A. Zemmouri, M. Alareqi,
Laboratory of Electrical Engineering and Energy System,
Faculty of Sciences, University Ibn Tofail,
Kenitra, Morocco.
Email: abdelkarim.zemmouri@gmail.com, alareqi_mohammed@yahoo.com

1. INTRODUCTION

The PWM is a digital signal, so the voltage can take only two values. In some very specific cases, they make a third level by reversing the high voltage [1]. All the necessary Xilinx hardware/software techniques and programs required to implement and generate pulse width modulation (PWM) are developed in detail, and demonstrated using practical results from an experimental Xilinx board [2],[3].

Using FPGA to generate the PWM provides flexibility to modify the designed circuit without change the hardware part. It also supports standard based Hardware description language deign. HDL configuration contains integrated VHDL synthesis and graphical interactive HDL entry tools. As we find in some application of FPGA Based PWM Control Technique for AC Motors [4].

Although the FPGA allow a high degree of parallelism in the implementation of PWM algorithms, it is necessary to reduce the total quantity of material needed, keeping the final cost to a reasonable point [5]. Most of the polarization circuits and embedded digital (MOS transistors, resistors, NE555, Arduino, etc.) can develop and generate a variable number of pulses, as well as the PWM, but they still limit in surface and in Input/output [6]-[8]. The low percentage of the device logic blocks occupied by the PWM implementation on the FPGA allows the integration of multiple control operations [9],[10]. Xilinx FPGA is a programmable logic device that is considered an effective hardware for rapid prototyping PWM generator [11].

The pulse width modulation has widely used in industrial control technology for regulating high-power circuits such as SVPWM (space-vector pulse-width modulation) [12]-[14] in the market [15]. Some studies were also introduces a design of a Micro Blaze soft core processor system that can be accommodated

to act as PWM system based on the implementation of blocks memories (SDRAM/DDR/DDR2 of 64 Mbytes) and timers/counters in the FPGA board [16].

The clocks are supply source of all blocks in digital processing. It is necessary in all embedded architectures in order to ensure the lines of frame pulse. The frequencies very high of these clocks provide a new approach for the control of the PWM in real time [17]. The remainder of this paper is divided into four parts. The first part describes the design MicroBlaze embedded type microprocessor that program in C. The second part methodology of work, all through the hardware and software environments. The third part simulation, results and discussion. Finally, concluding remarks are given in fourth part.

2. MICROBLAZE DESIGN

The MicroBlaze [18] is a virtual microprocessor that is built by combining blocks of code called cores inside a Xilinx Field Programmable Gate Array (FPGA).it is controlled by the system through the C language control instructions to operate the 100MHz main clock signals to create a PWM pulse (Figure 1). We can manipulate the period and the gap between the pulses. Therefore, we can control external circuits and drove them, motors, and high power sensors.

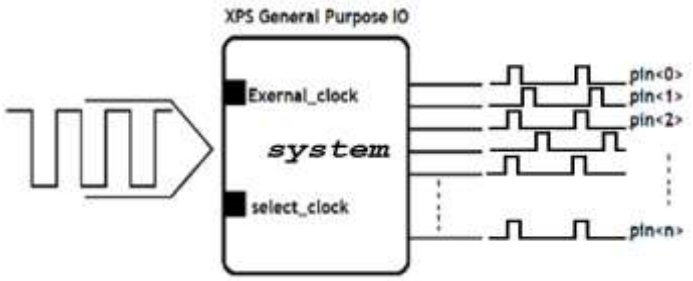

Figure 1. A methodology to multi-external pulse

The MicroBlaze processor (v8.10a) soft-core is highly configurable; it allows selecting a specific set of characteristics required for the design. The hardware part of the system is as shown in Figure 2, it is composed of:

a. 1 Processor Local Bus (PLB) 4.6.
b. 2 Local Memory Bus (LMB) 1.0 (I/DLMB for MAHB to 16 Kb).
c. 2 LMB BRAM Controller.
d. 1 XPS UART RS232 connector.
e. XPS General Purpose IO (8-pin extension (XGI Expansion Headers of 8-bit)).
f. 1 Clock Generator.
g. 1 MicroBlaze Debug Module (MDM).
h. 1 Processor System Reset Module.
a. The processor is targeted by a C code that provides architectural feature that includes:

i. 32-bit general-purpose registers.
j. 32-bit instruction word with three operands and two addressing modes.
k. 32-bit address bus.
l. Unique pipeline.

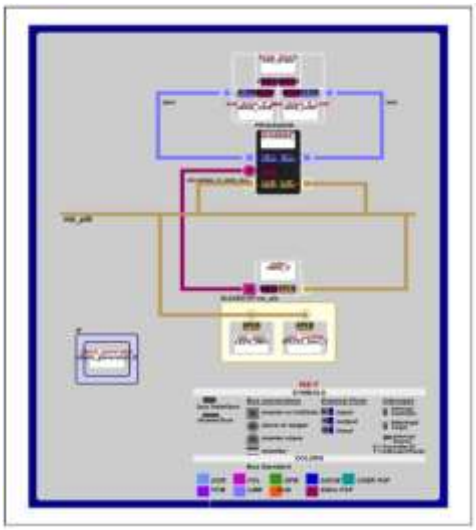

Figure 2. Block diagram of hardware in platform studio

In addition to these fixed characteristics, performance MicroBlaze processor (Figure 3) is set to allow selection of additional features. Only the latest version of MicroBlaze support all options.

Figure 3. Performance of microblaze (v8.10a) in the system

3. METHODOLOGY OF PROPOSED WORK

In the platforms Xilinx, the environment of XPS in EDK provided an infinite number of option for the creation hardware and choice of embedded components. Microblaze must be connected with its devices by PLBv46 connection; adding to the working space of a XPS output peripheral 8 bits for our demonstration (we can associate more than 8 bits depend the available of the card), in the finale files system.bit and system _bd.bmm is generated. Then these files was exported to the environment SDK to associate and compile with C. Eventually initialization ELF file in the RAM blocks is necessary to load the FPGA [2].

3.1. PWM hardware.

Virtex-5 device has 32 global clock lines that can support all sequential resources on the devices (PLB, block RAM, CMT, and I/O) and also drive logic signals. All ten of the 32 global clock lines can be used in all regions. Only global clock lines are driven by a slap global clock, which can also be used as a clock enable circuit.

A global clock buffer is often driven by a Clock Management Tile (CMT) to eliminate the clock distribution delay, or to adjust its delay relative to another clock. There are more global clocks than CMTs, but a CMT often drives more than one global clock [19]. Global clock buffers allow various clock/signal sources to access the global clock trees and nets. The possible sources for input to the global clock buffers include:

a. Global clock inputs
b. Clock Management Tile (CMT) outputs including:
c. Digital Clock Managers (DCMs)
d. Phase-Locked Loops (PLLs)
e. Other global clock buffer outputs
f. General interconnect

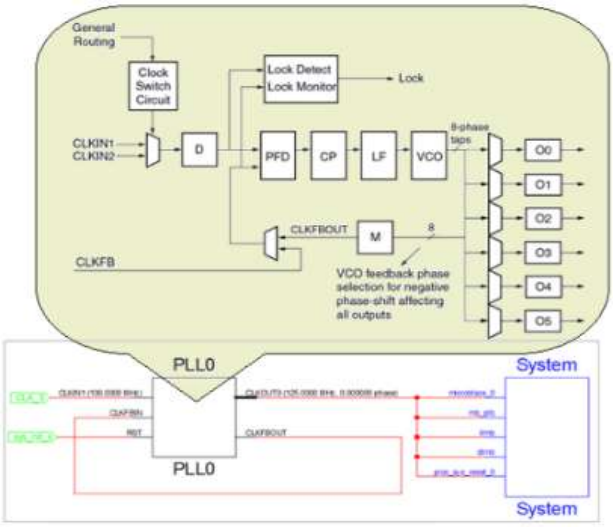

Figure 4. Detailed PLL clock block diagram in system schematics

The main purpose of PLL0 is to serve as a frequency synthesizer for a wide range of frequencies, and to serve as a filter for internal or external clocks, in conjunction with the RDR of the CMT. The module of phase-locked loop (PLL) is used to generate a system clock of 120MHz, i.e. 1CLK=8.33 ns, for FPGA based on an external oscillator of 100MHz [20].

3.2. PWM software

The system was operated through the C language program to give the PWM pulses. The game loops and instructions used to execute repeatedly the same series of instructions which in this case represents the outputs of the system states. Each instruction is repeated depending on the number N that can be chosen to determine the period T of each PWM pulse tram. N is the total number of executions of lines in the infinite loop.

The PWM is a digital signal, thus the voltage may take two values only. The signal is square, the low level generally corresponds to 0 Volt. Figure 5 shows N numbers compared to the cyclic ratio, it means that the time that the compiler moves from one line to another, it needs a Pseudo clock period to execute each line. "N" is the integer load to that under the loop increments while allowing the laundry to be active in the top or bottom level. The change of "n" causes a successive change of the period and the pulse width, which gives a good constant duty cycle; as these formulas describe:

$$\text{Pseudo period} = T / N \quad (1)$$

$$\alpha = \text{Duty Cycle} = (\text{Pulse Width} / \text{Period}) \times 100 \quad (2)$$

a. When Pulse Width = 0 then $\alpha = 0\%$ and the average output voltage is zero.
b. When Pulse Width = period then $\alpha = 100\%$ and the average output voltage is equal to Vcc.

The average voltage applied to the external circuit is proportional to the duty cycle:

$$\text{Average Voltage} = (\text{Pulse Width} \times Vcc) / \text{Period} \quad (3)$$

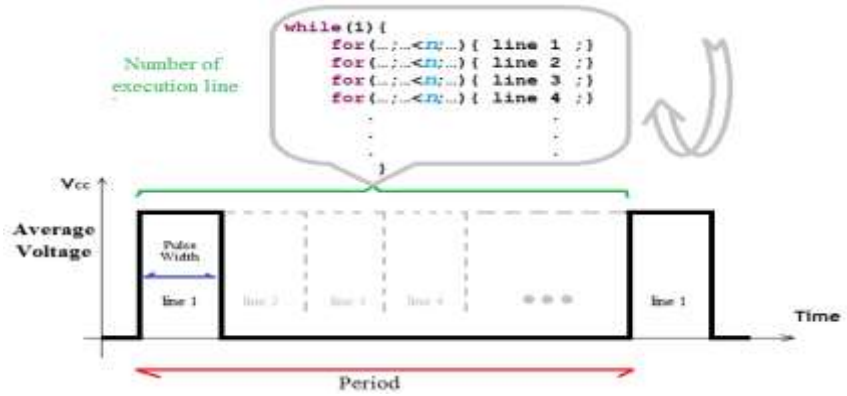

Figure 5. Pulse width modulated waveform

4. SIMULATION, RESULTS AND DISCUSSION

Figure 6 shows the Simulations on the ISim software; the pulse width varies in the same way for each of output signals, according to N and "n" that program. For this example was eight signals, which are activated by "1" or deactivate "0" according to the Table 1:

Table 1. Example of the output bits of each pin

Exernal_clock	On program
10000000	Line 1
01000000	Line 2
00100000	Line 3
00010000	Line 4
00001000	Line 5
00000100	Line 6
00000010	Line 7
00000001	Line 8

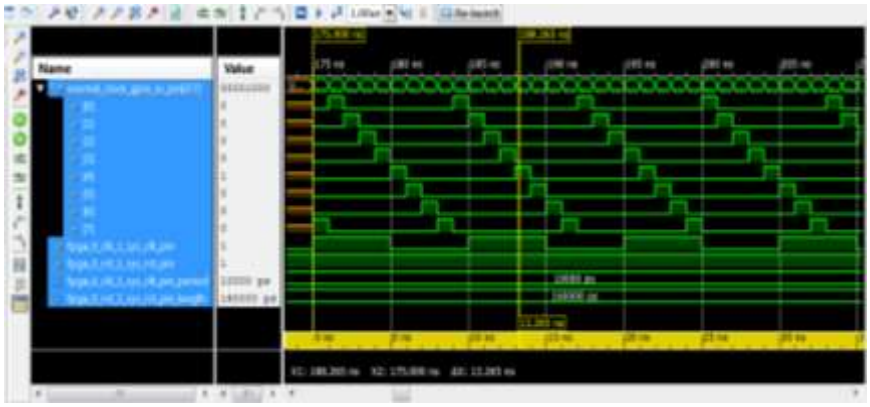

Figure 6. Timing waveforms

The Microblaze requires a local memory of 16 Kb (against 64 Mbytes for [16]) for that the program in C ensures the exchange of instruction. This embedded system will occupy spaces (Table 2) pretty weak in the FPGA chip as well as a low dissipation power (Table 3) by against the [16] added to other blocks memories, timers and counters that will also occupy enough space and consume enough powers.

Table 2. Space occupied by the circuit elements

Device Utilization Summary (actual values)			
Slice Logic Utilization	Used	Available	Utilization
Number of Slice Registers	2,058	69,120	2%
Number of Slice LUTs	2,107	69,120	3%
Number of occupied Slices	1,049	17,280	6%
Number of bonded IOBs	15	640	2%
Number of BlockRAM/FIFO	16	148	10%
Number of BUFG/BUFGCTRLs	2	32	6%
Number of BSCANs	1	4	25%
Number of DSP48Es	3	64	4%
Number of PLL_ADVs	1	6	16%

Table 3. The power dissipated by the pin outputs n = 8 lines

Signal Name	IO Bank Number	Drive (mA)	Voltage (mv)	Puissance (W)
Exernal_clock_GPIO_IO_pin<0-3>	11	12	472	5,664
Exernal_clock_GPIO_IO_pin<4-7>	13	12	472	5,664

The experimental result was display in the oscilloscope. This method complies with the equations (1-2 -3) which described by the PWM as it indicates the figures 6, 7 and 8 below. In case of two lines and n1 > n2 which gives a cyclic ratio of 75 %; in case of two lines and n1 = n2 which gives a cyclic ratio of 50 % and in the case of eight lines for a report of 13 %.

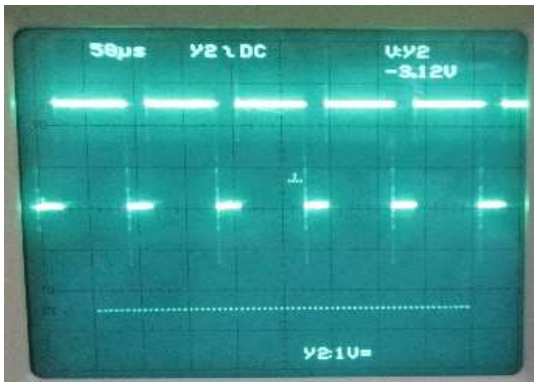

Figure 7. PWM α = 75% and N = 2

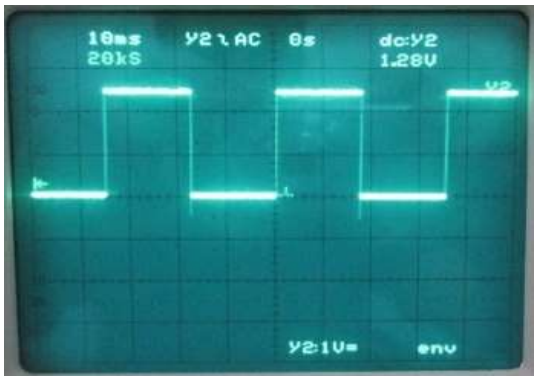

Figure 8. PWM α = 50% and N = 2

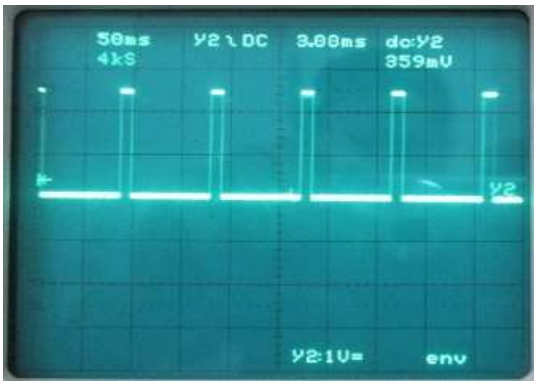

Figure 9. PWM α = 13% and N = 8

The cyclic ratio α is controlled in three phases:

a. For α = 50 % → n = constant, N = 2.
b. For α < 50 % → n = constant, N ↑.
c. For α > 50 % → n ↑, N = 2

4.1. Compere of results

4.1.1. FPGA-based embedded system

Xilinx proposed two methods to do the FPGA-based embedded system:

a. By VHDL which is a physical description of the circuit blocks (input/output, clock, process, component,...).
b. By a MicroBlaze embedded processor with its devices programmed in C.

These two methods rely on the FPGA as the LUTs, Registers, BUFG/BUFGCTRLs and input/output. Except that the PWM of the Microblaze circuit exceeds that of FPGA by some block as well as BSCANs, DSP48Es, PLL_ADVs, Memory used or Block RAM/FIFO.

The PLL_ADV block is used to generate a clock of 120 MHz, i.e. 1CLK = 8, 33 ns for the PWM system according to the external oscillator 100 MHz of FPGA and to serve as a frequency for a wide range of frequency synthesizer. The DSP48E receives a clock well synthesize by the PLL_ADV to speed up numeric calculations and ensure good quality results. These blocks embedded in the FPGA are well detailed in table 4 with their percentages of use in each circuit.

Table 4. Espace occupé par les éléments du circuit PWM, soit par l'implémentation en VHDL et en utilisant le processeur embarqué

Device Utilization Summary	VHDL			Microblaze		
Slice Logic Utilization	Used	Available	Utilization	Used	Available	Utilization
Number of Slice Registers	36	69,12	1%	1,649	69,12	2%
Number of Slice LUTs	38	69,12	1%	1,897	69,12	2%
Number used as logic	37	69,12	1%	1,754	69,12	2%
Number of occupied Slices	12	17,28	1%	938	17,28	5%
Number with an unused Flip Flop	3	39	7%	945	2,594	36%
Number with an unused LUT	1	39	2%	697	2,594	26%
Number of fully used LUT-FF pairs	35	39	89%	952	2,594	36%
Number of slice register sites lost to control set restrictions	4	69,12	1%	346	69,12	1%
Number of bonded IOBs	6	640	1%	12	640	1%
Number of LOCed IOBs	6	6	100%	12	12	100%
Number of BUFG/BUFGCTRLs	1	32	3%	2	32	6%
Number of BSCANs	--	--	--	1	4	25%
Number of DSP48Es	--	--	--	3	64	4%
Number of PLL_ADVs	--	--	--	1	6	16%
Number used as Memory	--	--	--	138	17,92	1%
Number of BlockRAM/FIFO	--	--	--	4	148	2%
Total Memory used (KB)	--	--	--	144	5,328	2%

4.1.2. Arduino-based embedded system

Arduino is a microcontroller board based on the ATmega640 / 1280/1281/2560/2561. The board can be programmed and configured by the Arduino Software IDE (Integrated Development Environment). This series microcontroller provides the following features: 64K / 128K / 256K bytes of In-System Programmable Flash with Read-While-Write capabilities, 4K bytes EEPROM, 8K bytes SRAM, six flexible Timer/Counters with compare modes and PWM (It has 54/86 digital pins of input / output 4/15 which can be used as PWM outputs) [21].

So the Arduino board does not have enough resources, which occupies less space compared to the FPGA (either by the MicroBlaze or alternatively by VHDL). However, In the case of complex projects, which includes the PWM, must use FPGA because it can support high-level algorithms by its memory capacity (internal and external memory). The power dissipated by the Arduino output pin is in the order of 165 mW, on the opposite of that FPGA output pin is about 5,664 mW. This large difference in energy consumption leads us to say that each card field of special use. The diversity of FPGA and low energy consumption help make the implementation of PWM circuit most reliable, flexible and good quality for the control of external circuits. Table 5 explains the energy balance of the PWM circuit of each card.

Table 5. La puissance dissipée par les broches de sorties PWM : FPGA/ARDUINO

Signal Name	Drive (mA)	Voltage (mV)	Puissance (mW)
PIN_OUT FPGA	12	472	5664
[21] PIN_OUT ARDUINO	50	3300	165000

The large power dissipation of the Arduino is explained by its cable assembly transmission lines, which consumes power when streaming data. On the other hand, FPGA architecture is purely on board which reduces the length of cables and consequently the dissipated energy.

5. CONCLUSION

Embedded circuits help much to improve the applications with their programs, such as the PWM, which does not occupy enough space in the FPGA architecture such as PWM not occupying enough space in the FPGA architecture, which operates a given memory or instruction 16 Kb, and a low power dissipation, as shown in the experimental results. This method allows users to take advantage of a number of outputs including the FPGA board has in order to increase the external applications manage by a single platform.

REFERENCES

[1] S. Mekhilef and N. A. Rahim, *"Xilinx FPGA based three-phase PWM inverter and its application for utility connected PV system,"* in TENCON'02. Proceedings. IEEE Region 10 Conference on Computers, Communications, Control and Power Engineering, vol. 3, pp. 2079-2082, 2002.

[2] A. Zemmouri, *et al.*, "Integration and Implimentation System-On-A-Programmable-Chip (Sopc) In FPGA," *Journal of Theoretical and Applied Information Technology*, vol. 76, pp. 127-133, 2015.

[3] Z. Zhou, *et al.*, "*Design of a universal space vector PWM controller based on FPGA,"* IEEE in Applied Power Electronics Conference and Exposition APEC'04. Nineteenth Annual, vol. 3, pp. 1698-1702, 2004.

[4] V. Stephen and P. Suresh, "Investigation of FPGA Based PWM Control Technique for AC Motors," *International Journal of Power Electronics and Drive Systems*, vol/issue: 3(2), pp. 193-199, 2013.

[5] Ó. López, *et al.*, "Comparison of the FPGA implementation of two multilevel space vector PWM algorithms," *IEEE Transactions on Industrial Electronics*, vol. 55, pp. 1537-1547, 2008.

[6] C. C. Wang, *et al.*, "A±3.07% frequency variation clock generator implemented using HV CMOS process," *Microelectronics Journal*, vol. 46, pp. 285-290, 2015.

[7] M. Ito, *et al.*, "Design and implementation of SFQ programmable clock generators," *Physica C: Superconductivity*, vol. 412, pp. 1550-1554, 2004.

[8] O. Cadenas and G. Megson, "A clocking technique for FPGA pipelined designs," *Journal of Systems Architecture*, vol. 50, pp. 687-696, 2004.

[9] E. Koutroulis, *et al.*, "High-frequency pulse width modulation implementation using FPGA and CPLD ICs," *Journal of Systems Architecture*, vol. 52, pp. 332-344, 2006.

[10] M. Facta, *et al.*, "The Application of FPGA in PWM Controlled Resonant Converter for an Ozone Generator," *International Journal of Power Electronics and Drive Systems (IJPEDS)*, vol/issue: 3(3), pp. 336-343, 2013.

[11] S. Mekhilef and A. Masaoud, "*Xilinx FPGA based multilevel PWM single phase inverter,"* IEEE International Conference in Industrial Technology ICIT, pp. 259-264, 2006.

[12] Y. Y. Tzou and H. J. Hsu, "FPGA realization of space-vector PWM control IC for three-phase PWM inverters," *IEEE Transactions on Power Electronics*, vol. 12, pp. 953-963, 1997.
[13] M. Saravanan, *et al.*, "Effectual SVPWM Techniques and Implementation of FPGA Based Induction Motor Drive," *International Journal of Reconfigurable and Embedded Systems*, vol/issue: 1(1), pp. 11-18, 2012.
[14] T. Sutikno, *et al.*, "FPGA Based Optimized Discontinuous SVPWM Algorithm for Three Phase VSI in AC Drives," *International Journal of Power Electronics and Drive Systems*, vol/issue: 3(2), pp. 228-240, 2013.
[15] A. Fratta, *et al.*, *"Comparative analysis among DSP and FPGA-based control capabilities in PWM power converters,"* 30th Annual Conference of IEEE in Industrial Electronics Society IECON, vol. 1, pp. 257-262, 2004.
[16] M. R. Khalil and S. B. Mahmood, "Designing Of A Pulse Width Modulation System Using Embedded System Design Techniques," *Journal of Theoretical and Applied Information Technology*, vol. 49, pp. 101-106, 2013.
[17] T. Yokoyama, *et al.*, *"Instantaneous deadbeat control for PWM inverter using FPGA based hardware controller*," The 29th Annual Conference of the IEEE in Industrial Electronics Society IECON'03, vol. 1, pp. 180-185, 2003.
[18] MicroBlaze Processor Reference Guide. www.xilinx.com. 2011 Mars; UG081 (v12.0).
[19] Virtex-5 FPGA User Guide. http://www.xilinx.com, http://www.xilinx.com/support/documentation/ip_documentation. 2008 December; 381 UG190 (v4.4).
[20] K. I. Hwu and Y. T. Yau, *"Applying a counter-based PWM control scheme to an FPGA-based SR forward converter,"* In Twenty-First Annual IEEE Applied Power Electronics Conference and Exposition APEC'06-09, pp. 5, 2006.
[21] P. Teikari, *et al.*, "An inexpensive Arduino-based LED stimulator system for vision research," *Journal of Neuroscience Methods*, vol/issue: 211(2), pp. 227– 236, 2012.